FPAI at SemEval-2021 Task 6: BERT-MRC for Propaganda Techniques Detection

Xiaolong Hou[*] **Junsong Ren**[*] **Gang Rao, Lianxin Jiang,**
Zhihao Ruan, Yang Mo, Jianping Shen
Ping An Life Insurance, Lt
ShenZhen, China
{houxiaolong430, renjunsong941, raogang532, jianglianxin769, ruanzhihao332, moyang853, shenjianping324}@pingan.com.cn

Abstract

The objective of subtask 2 of SemEval-2021 Task 6 is to identify techniques used together with the span(s) of text covered by each technique. This paper describes the system and model we developed for the task. We first propose a pipeline system to identify spans, then to classify the technique in the input sequence. But it severely suffers from handling the overlapping in nested span. Then we propose to formulize the task as a question answering task by machine reading comprehension (MRC) framework which achieves a better result compared to the pipeline method. Moreover, data augmentation and loss design techniques are also explored to alleviate the problem of data sparsity and imbalance. Finally, we attain the 3^{rd} place in the final evaluation phase.

1 Introduction

Fake news detection has attracted the attention of many researchers. But most of the conventional fake news detection methods focuse on long-form journalism, and little attention has been paid to the propaganda techniques. Memes have become the most popular type of content on social media platforms, and it can easily mislead the audience to agree with the speaker through propaganda techniques. The SemEval-2021 Task 6 focuses on detecting the use of rhetorical and psychological techniques in memes without (subtask 1, subtask 2) and with (subtask 3) visual content.

We first adopted a model to identify the span and techniques sequentially, and made many attempts to optimize the model, but the results were not satisfactory. Then we tried some end-to-end method, e.g. MRC framework.

[*]These authors contributed equally to this work This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http:// creativecommons.org/licenses/by/4.0/

We found that the pipeline model can't handle the span overlapping issue effectively, but the MRC framework with an informative query is well performing in this scenario. In addition, the data augmentation and a carefully designed loss can alleviate the impact of data sparsity and imbalance. We attain an F1 score of 0.3974 and the 3^{rd} place in the final evaluation phase

2 Related Work

There was also a related previous task on fine-grained propaganda detection (Da San Martino et al., 2019), where the participants used Transformer-style models, LSTMs and ensembles (Fadel et al.,2019;Hou and Chen,2019;Hua,2019). Some approaches further used non-contextualized word embeddings, e.g., based on FastText and GloVe (Gupta et al.,2019; Al-Omari et al., 2019), or handcrafted features such as LIWC, quotes and questions (Alhindi et al., 2019). Moreover, Martino et al.2020 analysed computational propaganda detection from Text Perspective and Network Perspective, argued for the need of combined efforts blending Natural Language Processing, Network Analysis, and Machine Learning.

3 System Description

3.1 Pipeline model for Span and Technique Detection

Inspired by the method (Chernyavskiy et al., 2020) proposed for NER task, we construct a pipeline model with RoBERTa(Liu et al., 2019) as the backbone to identify spans and techniques in input sequence. Figure 1 depicts our proposed pipeline model.

3.1.1 Span Identification

We treat the span identification as a binary sequence tagging task. The span identification model

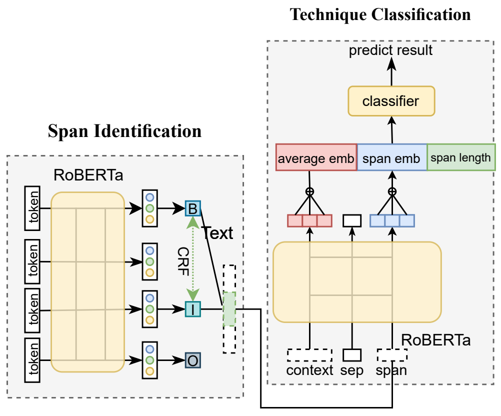

Figure 1: Pipeline Model for Span and Technique Detection

is fed with chunks of sentences encoded by Byte-Pair-Encoding (BPE) and the Conditional Random Field (CRF) layer (Lafferty et al., 2001) receives the logits for each input token, and makes a BIO prediction for the entire input sequence, finally got the spans in input sequence.

3.1.2 Technique Classification

We take the result of span identification model as a part of the input: `[CLS]<context>[SEP]<span>`, where `<context>` is the sentence from which the span is extracted. The input of softmax layer includes three parts, *(i)* context embedding, is extracted from the last two layers of RoBERTa model; *(ii)* span embedding, is the average of span tokens embedding; *(iii)* span length embedding, is constructed with the length of the span, as different propaganda techniques have significant differences in span length. Finally, the model output the technique category for the given input sequence.

As the pipeline model suffers from incapable of handling overlapping issue in nested span, significantly inspired by (Li et al., 2019), we proposed to utilize the MRC framework to identify the span and corresponding techniques.

3.2 Span Detection as MRC

We formulize the task as a question answering task. Each span is characterized by a query, and span are extracted by answering these queries given the context. For example, the task of assigning the `Name calling/Labeling` to *"CALM DOWN [LITTLE TRUMP HATER]\nI FOUND YOUR BINKY\n "* is formalized as answering the question *"Find the words and phrases with strong emotional implications (either positive or negative) that influence an audience"*. This strategy naturally tackles the span overlapping issue in nested span: the detection of different spans that overlap requires answering different independent questions.

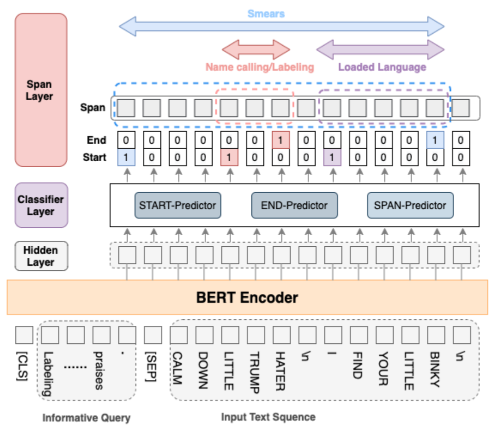

Figure 2: BERT-MRC Span Detection Model

3.2.1 BERT Encoder

Given the question q_y, we need to extract the text span $x_{start,end}$ from input text sequence $X = \{x_1, x_2, ...x_n\}$ given q_y using MRC frameworks. We use BERT as encoder with the concatenated informative query q_y and input sequence X as input by adding special token [CLS] and [SEP], and get the whole representation matrix H from BERT.

$$H = BERT([q_y, X]) \tag{1}$$

3.2.2 START-Predictor

The START-Predictor output the probability of each token being a start token of a span, and E_{start} is the parameter to learn:

$$P_{start} = softmax(H \cdot E_{start}) \tag{2}$$

3.2.3 END-Predictor

The END-Predictor output the probability of each token being an end token of a span, E_{end} is the parameter to learn:

$$P_{end} = softmax(H \cdot E_{end}) \tag{3}$$

3.2.4 SPAN-Predictor

As there could exist more than one span in given input sequence, we need to predict multiple start token and end token. Deciding which start-end pair that consist of continuous token sequence between start token and end token. We get the possible start token indexes I_{start} and end token indexes I_{end} by applying $argmax$ on P_{start} and P_{end}. Each i in

I_{start} and j in I_{end} construct a continuous token sequences $x_{i,j}$ with constriction $i \leq j$. A binary classifier is used to compute the probability of $x_{i,j}$ as a valid span, and E_{span} is the parameter to learn:

$$I_{start} = argmax(P_{start}) \quad (4)$$
$$I_{end} = argmax(P_{end}) \quad (5)$$
$$p_{i,j} = sigmoid(E_{span} \cdot [E_i; E_j]) \quad (6)$$

3.2.5 Train

During the training stage, input sequence X is with two label sequence Y_{start} and Y_{end}, representing the ground-truth label of each token x_i being the start index and end index. $Y_{start,end}$ the ground-truth span. And the loss for each predictor as follows:

$$L_{start} = CrossEntropy(P_{start}, Y_{start}) \quad (7)$$
$$L_{end} = CrossEntropy(P_{end}, Y_{end}) \quad (8)$$
$$L_{span} = CrossEntropy(P_{span}, Y_{span}) \quad (9)$$

The model is end-to-end fine-tuned by minimizing the loss as follows:

$$L = L_{start} + L_{end} + L_{span} \quad (10)$$

3.2.6 Predict

For every example to be predicted, we duplicate the example 20 times and pair each example with one technique description as model input. The model output spans detected for each example, and techniques can be mapped from those examples with spans detected.

4 Experiment and Result

We use the provided training dataset for training, and evaluate the model on the development dataset during evaluation phase. We implemented several experiments to explore the effect of different methods. The following is the detail for each experiment.

4.1 Data augmentation

As there are only 290 records in the training dataset, we explored two data augmentation methods to increase the amount of relevant data. We did the data augmentation based on PTC corpus[1]. And Enlarge Training Dataset get a better result.

[1] https://propaganda.qcri.org/semeval2020-task11/

Data augmentation	**F1**
Two-Stage Fine-Tune	0.19322
Enlarge Training Dataset	**0.21053**
Train Dataset	0.19073

Table 1: Result for Data Augmentation.

4.1.1 Two-Stage Fine-Tune

Train a best-performing model using BERT as backbone with PTC corpus, then finetune the trained model on the training dataset. Table 1 shows the result of this method.

4.1.2 Enlarge Training Dataset

Train a best-performing model using BERT as backbone with training dataset and annotate PTC corpus automatically. After that we train the second model using the filtered samples according to annotated labels and training dataset. Table 1 shows the result of this method.

4.2 Loss setup

This task is faced with serve data imbalance issue: the top 6 technique category examples account for 78%, which is significantly outnumber the reset of 14 technique category examples, leading to huge number of top 6 examples overwhelms training. In order to remit the impact of imbalance issue, we tried different loss functions. Table 2 shows the result of different loss function. From the result we can see that Asymmetric Loss improved the result over other loss function.

4.2.1 Focal Loss

By setting $\gamma \geq 0$ in Eq.12, the contribution of easy samples can be down-weighted in the loss function, enabling to focus more on harder samples during training(Lin et al., 2018).

$$L = -yL_+ - (1-y)L_- \quad (11)$$

$$\begin{cases} L_+ = (1-p)^\gamma \log p \\ L_- = p^\gamma \log(1-p) \end{cases} \quad (12)$$

4.2.2 Asymmetric Loss

Asymmetric loss can perform hard thresholding of very easy samples, meaning fully discard negative samples when their probability is low enough(Ben-

Loss Function	**F1**
CrossEntropy	0.19073
Focal Loss	0.20453
Asymmetric Loss	**0.20817**

Table 2: Result for Different Loss.

Model	**F1**
BERT-MRC$_{neg=2}$	0.2986
BERT-MRC$_{neg=4}$	0.3665
BERT-MRC$_{neg=10}$	0.3463
BERT-MRC$_{neg=4\_DA}$	**0.3779**
BERT-MRC	0.2815

Table 3: Results for MRC-BERT.

Baruch et al., 2020).

$$L = -yL_+ - (1-y)L_- \tag{13}$$

$$p_m = \max(p-m, 0) \tag{14}$$

$$ASL = \begin{cases} L_+ = (1-p)^{\gamma_+} \log p \\ L_- = p_m^{\gamma_-} \log(1-p_m) \end{cases} \tag{15}$$

Where the probability margin $m \geq 0$ is a tunable hyperparameter.

4.3 BERT-MRC Span Detection

In the training dataset, some examples do not contain any technology. We duplicate each example 20 times and pair same example with different technique description as model input. The example paired with the technique it contained is treated as positive examples and others as negative examples. We also tried to sample different number of negative examples. Table 3 shows the result of different sampling strategy. When sample 4 negative examples, the model BERT-MRC$_{neg=4\_DA}$ achieve the best result. The performance decrease when sample more negative examples, which may be caused by introducing too much noise into the model. The model is trained with hyperparameter learning rate 5e-5, batch size 8, max sequence length 128, and bert-base-cased as the backbone.

5 Conclusion

We create a pipeline model and an end-to-end MRC model to identify the span and techniques. We attain an F1 score of 0.3974 and the 3^{rd} place at in final evaluation phase with BERT-MRC$_{neg=4\_DA}$ model. Our implementation shows that 1) reformulating the task into an MRC task to detect the overlapping span is effective; 2) data augmentation is about to increase model generalization ; 3) careful loss design can alleviate the effect of data imbalance.

References

Hani Al-Omari, Malak Abdullah, Ola AlTiti, and Samira Shaikh. 2019. JUSTDeep at NLP4IF 2019 task 1: Propaganda detection using ensemble deep learning models. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 113–118, Hong Kong, China. Association for Computational Linguistics.

Tariq Alhindi, Jonas Pfeiffer, and Smaranda Muresan. 2019. Fine-tuned neural models for propaganda detection at the sentence and fragment levels. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 98–102, Hong Kong, China. Association for Computational Linguistics.

Emanuel Ben-Baruch, Tal Ridnik, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. 2020. Asymmetric loss for multi-label classification.

Anton Chernyavskiy, Dmitry Ilvovsky, and Preslav Nakov. 2020. Aschern at SemEval-2020 task 11: It takes three to tango: RoBERTa, CRF, and transfer learning. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 1462–1468, Barcelona (online). International Committee for Computational Linguistics.

Giovanni Da San Martino, Alberto Barrón-Cedeño, and Preslav Nakov. 2019. Findings of the NLP4IF-2019 shared task on fine-grained propaganda detection. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 162–170, Hong Kong, China. Association for Computational Linguistics.

Ali Fadel, Ibraheem Tuffaha, and Mahmoud Al-Ayyoub. 2019. Pretrained ensemble learning for fine-grained propaganda detection. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 139–142, Hong Kong, China. Association for Computational Linguistics.

Pankaj Gupta, Khushbu Saxena, Usama Yaseen, Thomas Runkler, and Hinrich Schütze. 2019. Neural architectures for fine-grained propaganda detection in news. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 92–97, Hong Kong, China. Association for Computational Linguistics.

Wenjun Hou and Ying Chen. 2019. CAUnLP at NLP4IF 2019 shared task: Context-dependent BERT for sentence-level propaganda detection. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 83–86, Hong Kong, China. Association for Computational Linguistics.

Yiqing Hua. 2019. Understanding BERT performance in propaganda analysis. In *Proceedings of the Second Workshop on Natural Language Processing for Internet Freedom: Censorship, Disinformation, and Propaganda*, pages 135–138, Hong Kong, China. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. pages 282–289.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2019. A unified MRC framework for named entity recognition. *CoRR*, abs/1910.11476.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2018. Focal loss for dense object detection.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Giovanni Da San Martino, Stefano Cresci, Alberto Barron-Cedeno, Seunghak Yu, Roberto Di Pietro, and Preslav Nakov. 2020. A survey on computational propaganda detection.